IFW

OIPE JUN 0 8 2004

PTO/SB/21 (08-00)
Approved for use through 10/31/02. OMB 0651-0031
Patent and Trademark Office: U.S. DEPARTMENT OF COMMENRCE
Under the Paper Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays avalid OMB control number.

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Application Number	10/609,212
Filing Date	June 27, 2003
First Named Inventor	Didier Poirot, et al.
Group Art Unit	2153
Examiner Name	NYA
Attorney Docket Number	15437-0628
Total Number of Pages in This Submission	27

ENCLOSURES *(check all that apply)*

- [ ] Fee Transmittal Form
  - [ ] Fee Attached
- [ ] Amendment / Response
  - [ ] After Final
  - [ ] Affidavits/declaration(s)
- [ ] Extension of Time Request
- [ ] Express Abandonment Request
- [ ] Information Disclosure Statement
- [x] Certified Copy of Priority Document(s) from France dtd 27 Mars 2003
- [ ] Response to Missing Parts/ Incomplete Application
  - [ ] Response to Missing Parts under 37 CFR 1.52 or 1.53
- [ ] Assignment Papers *(for an Application)*
- [ ] Drawing(s)
- [ ] Licensing-related Papers
- [ ] Petition
- [ ] Petition To Convert To a Provisional Application
- [ ] Power of Attorney, Revocation Change of Correspondence Address
- [ ] Terminal Disclaimer
- [ ] Request for Refund
- [ ] CD, number of CD(s)______
- [ ] After Allowance Communication to Group
- [ ] Appeal Communication to Board of Appeals and Interferences
- [ ] Appeal Communication to Group *(Appeal Notice, Brief, Reply Brief)*
- [ ] Proprietary Information
- [ ] Status Letter
- [ ] Other Enclosure(s) *(please identify below)*:

Remarks

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm *or* Individual name	Hickman Palermo Truong & Becker LLP Craig G. Holmes
Signature	[signature]
Date	June 3, 2004

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on this date: 6/3/04

Type or printed name	Annette Jacobs		
Signature	[signature]	Date	June 3, 2004

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time your are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

P7271

R E P U B L I Q U E F R A N Ç A I S E

2 08078
letter 1

INPI
INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE

BREVET D'INVENTION

CERTIFICAT D'UTILITÉ - CERTIFICAT D'ADDITION

COPIE OFFICIELLE

Le Directeur général de l'Institut national de la propriété industrielle certifie que le document ci-annexé est la copie certifiée conforme d'une demande de titre de propriété industrielle déposée à l'Institut.

Fait à Paris, le 27 MARS 2003

Pour le Directeur général de l'Institut
national de la propriété industrielle
Le Chef du Département des brevets

Martine PLANCHE

INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE
SIEGE
26 bis, rue de Saint Petersbourg
75800 PARIS cedex 08
Téléphone : 33 (0)1 53 04 53 04
Télécopie : 33 (0)1 53 04 45 23
www.inpi.fr

ETABLISSEMENT PUBLIC NATIONAL CREE PAR LA LOI N° 51-444 DU 19 AVRIL 1951

1er dépôt

INPI
INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE

26 bis, rue de Saint Pétersbourg
75800 Paris Cedex 08
Téléphone : 01 53 04 53 04 Télécopie : 01 42 94 86 54

BREVET D'INVENTION
CERTIFICAT D'UTILITÉ

cerfa N° 11354*01

Code de la propriété intellectuelle - Livre VI

REQUÊTE EN DÉLIVRANCE 1/2

Important ! Remplir impérativement la 2ème page.

Cet imprimé est à remplir lisiblement à l'encre noire

DB 540 W /190600

Réservé à l'INPI

REMISE DES PIÈCES DATE	28 JUIN 2002
LIEU	75 INPI PARIS
N° D'ENREGISTREMENT NATIONAL ATTRIBUÉ PAR L'INPI	0208078
DATE DE DÉPÔT ATTRIBUÉE PAR L'INPI	28 JUIN 2002

1 NOM ET ADRESSE DU DEMANDEUR OU DU MANDATAIRE À QUI LA CORRESPONDANCE DOIT ÊTRE ADRESSÉE

CABINET NETTER
36 avenue Hoche
75008 PARIS

Vos références pour ce dossier *(facultatif)* SUN Aff. 33 (120678)

Confirmation d'un dépôt par télécopie ☐ N° attribué par l'INPI à la télécopie

2 NATURE DE LA DEMANDE	**Cochez l'une des 4 cases suivantes**	
Demande de brevet	☒	
Demande de certificat d'utilité	☐	
Demande divisionnaire	☐	
Demande de brevet initiale	N°	Date / /
ou demande de certificat d'utilité initiale	N°	Date / /
Transformation d'une demande de brevet européen *Demande de brevet initiale*	☐ N°	Date / /

3 TITRE DE L'INVENTION (200 caractères ou espaces maximum)

Localization of computers in distributed computer system.

4 DÉCLARATION DE PRIORITÉ OU REQUÊTE DU BÉNÉFICE DE LA DATE DE DÉPÔT D'UNE DEMANDE ANTÉRIEURE FRANÇAISE	
	Pays ou organisation Date / / N°
	Pays ou organisation Date / / N°
	Pays ou organisation Date / / N°
	☐ **S'il y a d'autres priorités, cochez la case et utilisez l'imprimé «Suite»**

5 DEMANDEUR		☐ **S'il y a d'autres demandeurs, cochez la case et utilisez l'imprimé «Suite»**
Nom ou dénomination sociale		SUN MICROSYSTEMS, INC
Prénoms		
Forme juridique		
N° SIREN		
Code APE-NAF		
Adresse	Rue	901 San Antonio Road
	Code postal et ville	94303 PALO ALTO Californie
Pays		Etats-Unis d'Amérique
Nationalité		Société des Etats-Unis d'Amérique
N° de téléphone *(facultatif)*		
N° de télécopie *(facultatif)*		
Adresse électronique *(facultatif)*		

1er dépôt

INPI
INSTITUT NATIONAL DE LA PROPRIÉTÉ INDUSTRIELLE

BREVET D'INVENTION
CERTIFICAT D'UTILITÉ

REQUÊTE EN DÉLIVRANCE 2/2

Réservé à l'INPI

REMISE DES PIÈCES
DATE 28 JUIN 2002
LIEU 75 INPI PARIS
N° D'ENREGISTREMENT NATIONAL ATTRIBUÉ PAR L'INPI 0208078

DB 540 W /190600

Vos références pour ce dossier : *(facultatif)*	SUN Aff. 33 (120678)
6 MANDATAIRE	
Nom	BEZAULT
Prénom	Jean
Cabinet ou Société	Cabinet NETTER
N °de pouvoir permanent et/ou de lien contractuel	
Adresse — Rue	36 avenue Hoche
Adresse — Code postal et ville	75008 PARIS
N° de téléphone *(facultatif)*	01 58 36 44 22
N° de télécopie *(facultatif)*	01 42 25 00 45
Adresse électronique *(facultatif)*	
7 INVENTEUR (S)	
Les inventeurs sont les demandeurs	☐ Oui ☒ Non **Dans ce cas fournir une désignation d'inventeur(s) séparée**
8 RAPPORT DE RECHERCHE	**Uniquement pour une demande de brevet (y compris division et transformation)**
Établissement immédiat ou établissement différé	☐ ☒
Paiement échelonné de la redevance	**Paiement en deux versements, uniquement pour les personnes physiques** ☐ Oui ☐ Non
RÉDUCTION DU TAUX DES REDEVANCES	**Uniquement pour les personnes physiques** ☐ Requise pour la première fois pour cette invention *(joindre un avis de non-imposition)* ☐ Requise antérieurement à ce dépôt *(joindre une copie de la décision d'admission pour cette invention ou indiquer sa référence) :*
Si vous avez utilisé l'imprimé «Suite», indiquez le nombre de pages jointes	

10 SIGNATURE DU DEMANDEUR OU DU MANDATAIRE (Nom et qualité du signataire) N° Conseil 92-1024 (B) (M) Jean BEZAULT	**VISA DE LA PRÉFECTURE OU DE L'INPI** M. ROCHET

La loi n°78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés s'applique aux réponses faites à ce formulaire. Elle garantit un droit d'accès et de rectification pour les données vous concernant auprès de l'INPI.

sun33.FRD.wpd
P7271

Localization of computers in distributed computer system

The invention relates to a distributed computer system comprising computers or other hardware entities called nodes.

Some distributed computer systems comprise connection entities such as switches to establish connection between nodes. In distributed computer systems, especially in distributed computer systems requiring to be highly available, it is highly important to improve communications between nodes. Thus, some nodes require to exchange a great number of messages. For these nodes, it is of particular interest to reduce network distances between them and to gather these nodes in connecting them on the same connection entity or on neighbor connection entity. These requirements involve node localization. Other improvements of node management may be performed when a node localization is provided.

A general aim of the present invention is to provide advances in this matter.

The invention concerns a distributed computer system, comprising :
- a plurality of nodes,
- a switch having ports,
- each port having an identifier,
- each node having an identifier and some nodes being connected to ports of the switch,
- the switch comprising an agent code arranged for accessing to port status and identifiers of nodes connected to ports,
- a node in said plurality comprising a manager code arranged for
    - retrieving the status of a port from the agent code and, responsive to said status meeting a given condition, retrieving the identifier of the node connected to said port from the agent code,
    - maintaining a table of data groups comprising identifier of a node connected to a port and the identifier of said port.

The invention also concerns a method of managing a localization of distributed computer system comprising a plurality of nodes and a switch having ports, each port having an

α

identifier, each node having an identifier and being connected to a port of the switch, the switch having access to port status and identifiers of nodes connected to ports, comprising the following steps from said node:

a. retrieving the status of a port from the switch

b. responsive to said status meeting a given condition, retrieving the identifier of the node connected to said port from the switch, and

c. maintaining a table of data groups comprising port identifier and the identifier of the node connected to said port.

Other alternative features and advantages of the invention will appear in the detailed description below and in the appended drawings, in which :

- figure 1 is a general diagram of a node in a distributed computer system;

- figure 2 is a general diagram of a distributed computer system comprising nodes connected using switches;

- figure 3 is a functional diagram of a node using a information management protocol on network, e.g. SNMP;

- figure 4 is a functional diagram of a switch using an information management protocol on network, e.g. SNMP;

- figure 5 is a table of data groups comprising identifiers of switch port linked to identifiers of node according to an embodiment of the invention;

- figure 6 is a flow-chart illustrating the method to build the table of figure 5 according to an embodiment of the invention;

- figure 7 is a flow-chart illustrating the method to update the table of figure 5 according to an embodiment of the invention.

α

Additionally, the detailed description is supplemented with the following Exhibits:
- Exhibit I contains pseudo-code of functions used in the SNMP protocol.

These Exhibits are placed apart for the purpose of clarifying the detailed description, and of enabling easier reference. They nevertheless form an integral part of the description embodiments of the present invention. This applies to the drawings as well.

This invention also encompasses embodiments in software code, especially when made available on any appropriate computer-readable medium. The expression "computer-readable medium" includes a storage medium such as magnetic or optic, as well as a transmission medium such as a digital or analog signal.

This invention may be implemented in a network comprising computer systems. The hardware of such computer systems is for example as shown in Fig. 1, where in the computer system Ni:

- 1 is a processor, e.g. an Ultra-Sparc (SPARC is a Trademark of SPARC International Inc);
- 2 is a program memory, e.g. an EPROM for BIOS;
- 3 is a working memory for software, data and the like, e.g. a RAM of any suitable technology (SDRAM for example); and
- 7 is a network interface device connected to a communication medium 8, itself in communication with a switch to enable communication with other computers. Network interface device 7 may be an Ethernet device, a serial line device, or an ATM device, inter alia. Medium 8 may be based on wire cables, fiber optics, or radio-communications, for example.

The computer system, also called node Ni, may be a node amongst a group of nodes in a distributed computer system. Some nodes may further comprise a mass memory, e.g. one or more hard disks.

Data may be exchanged between the components of Figure 1 through a bus system 9, schematically shown as a single bus for simplification of the drawing. As is known, bus systems may often include a processor bus, e.g. of the PCI type, connected via appropriate bridges to e.g. an ISA bus and/or an SCSI bus.

α

Figure 2 shows an example of a group of nodes arranged as a cluster. The cluster has several nodes N1, N2, N3, N4, N5, ... N10.

References to the drawings in the following description will use two different indexes or suffixes i and j, each of which may take anyone of the values: {1, 2, 3...,n} n being the number of nodes in the cluster. In the foregoing description, a switch is only an example of a connection entity for nodes on the network.

In figure 2, each node Ni is connected to a network, e.g. the Ethernet network which may be also the internet network. The node Ni is firstly connected to a switch SA, e.g. an Ethernet switch, capable of interconnecting the node Ni with other nodes Nj. The switch comprises several ports P, each being capable of connecting a node Ni to the switch SA via a link L. In an embodiment of a switch, the number of ports per switch is limited, e.g. to 24 ports in some switch technologies. Several switches may be linked together in order to increase the number of nodes connected to the network, e.g. the ethernet network. Thus, in figure 2, a switch SB is connected to the switch SA via a link E, e.g. an ethernet link. By way of example only, the switch may be called an ethernet switch if the physical network is an ethernet network. Indeed, different switch types exist such as ethernet switch and internet switch also called IP switch. Each switch has an identifier :

- for an ethernet switch, the identifier is e.g. a MAC address being an ethernet address or an IP address for administration,
- for an IP switch, the identifier is e.g. an IP address.

Each switch port has an identifier, e.g. a port number being generally an integer or an ethernet port address.

In the following description, an ethernet switch is used but the invention is not restricted to this switch type.

If desired, for availability reasons, the network may be also redundant, e.g. the ethernet network. Thus, the links L may be redundant: nodes Ni of the cluster are connected to a second network via links L' using a redundant switch as a switch SA' (not shown on figure 2). This redundant network is adapted to interconnect a node Ni with another node Nj through the links L'. For example, if node Ni sends a packet to node Nj, the packet may be

therefore duplicated to be sent on both networks. In fact, this redundancy may not be further described in the foregoing description, but the second network for a node may be used in parallel with the first network or replace it in case of first network failure.

Also, as an example, it is assumed that packets are generally built throughout the network in accordance with a transport protocol and a presentation protocol, e.g. the Ethernet Protocol and the Internet Protocol. Corresponding IP addresses are converted into Ethernet addresses on Ethernet network.

A node is connected to other nodes of a group of nodes (cluster) using a connection entity as the switch. When an administrator connects a node to the group of nodes, he connects physically the node to the network but he has no idea to which switch and to which port the node is connected. Thus, once connected to a port of a switch, the node is not localized on the network. The invention provides improvements in this matter.

Figure 3 shows an exemplary node Ni, in which the invention may be applied. Node Ni comprises, from top to bottom, applications 13, management layer 11, network protocol stack 10, and Link level interface 12, and optionally Link level interface 14 in case of network redundancy, connected to first network with link L1, respectively to second network with link L2. Applications 13 and management layer 11 can be implemented, for example, in software executed by the node's CPU. Network protocol stack 10 and link level interfaces 12 and 14 can likewise be implemented in software and/or in dedicated hardware such as the node's network hardware interface 7 of figure 1. Node Ni may be part of a local or global network; in the foregoing exemplary description, the network is an Ethernet network, by way of example only. It is assumed that each node may be uniquely defined by a portion of its Ethernet address. Accordingly, as used hereinafter, "IP address" means an address uniquely designating a node in the network being considered (e.g. a cluster), whichever network protocol is being used. Although Ethernet is presently convenient, no restriction to Ethernet is intended.

Thus, in the example, network protocol stack 10 comprises:

- an IP interface 100, having conventional Internet protocol (IP) functions 102,

- above IP interface 100, message protocol processing functions, such as UDP function 104 and TCP function 106.

Network protocol stack 10 is interconnected with the physical networks through first and second Link level interfaces 12 and 14, respectively and if network redundancy is desired. These are in turn connected to first and second network channels, via couplings L1 and L2 and via first and second switches.

Link level interface 12 has an Internet address <*IP_12*> and a link level address <<*LL_12*>>. Incidentally, the doubled triangular brackets (<< ... >>) are used only to distinguish link level addresses from global network addresses. Similarly, Link level interface 14 has an Internet address <*IP_14*> and a link level address <<*LL_14*>>. In a specific embodiment, where the physical network is Ethernet-based, interfaces 12 and 14 are Ethernet interfaces, and <<*LL_12*>> and <<*LL_14*>> are Ethernet addresses.

IP functions 102 comprise encapsulating a message coming from upper layers 104 or 106 into a suitable IP packet format, and, conversely, de-encapsulating a received packet before delivering the message it contains to upper layer 104 or 106.

An interface may be adapted in the IP interface 100 to manage redundancy of packets from the link level interfaces 12 and 14.

References to Ethernet are exemplary and other physical network may be used, implying Link level interfaces 12 and 14 based on other network. Moreover, other protocols than TCP or UDP may be used as well in stack 10.

The node comprises in its application layer 13 a manager module 130-M adapted to :
- use a network information management protocol, e.g. the simple network management protocol (SNMP) as described in RFC 1157 (may 1990), in order to work in relation with an agent module, specifically an agent module in a connection entity, e.g. a switch, using advantageously the same network information management protocol as described in figure 4,

- request an agent module to perform network management functions defined by the SNMP protocol (see exhibit I-1), such as a *get-request(var)* function requesting the agent module to return the value of the requested variable *var*, a *get-next-request(var)* function requesting the agent module to return the next value associated with a variable e.g. a table that contains a list of elements, the *set-request(var,val)* function requesting the agent module to set the value *val* of the requested variable *var*.

The manager module 130-M is linked to a memory 105 in order to store data being at least information retrieved from an agent module. The SNMP protocol used in the manager module 130-M may be based on the UDP/IP transport protocol or other transport protocols such as TCP/IP.

Figure 4 shows a switch SI adapted to connect nodes between them and also to be connected to other switches. The switch of figure 4, being e.g. an ethernet switch, comprises an agent module 130-A. This agent module is adapted to

- use a network information management protocol, e.g. the simple network management protocol (SNMP),
- perform network management functions (see exhibit I-1) requested by nodes having a manager module 130-M and
- transmit results of requests with the *get-response(var)* function or transmit exceptional events to said manager module 130-M with the *trap(code)* function (see exhibit I-2) .

The simple network management protocol (SNMP) is a management protocol at an application level as described in the RFC 1157 (may 1990). It is based on a network protocol stack 10-S comprising IP stack 102-S and message protocol processing functions, e.g. an UDP function 104 and/or a TCP function 106. The SNMP protocol used in the agent module 130-A may be based on the UDP/IP transport protocol or other transport protocols such as TCP/IP.

The manager module and the agent module may also be designated as a manager code and an agent code.

The SNMP protocol enables an agent module to implement several Management Information Bases (MIB) used by the manager module. A MIB interface concerns

λ

configuration information for devices, e.g. the definition of paper sheet dimensions for printing devices, a MIB switch concerns information on the connections between the ports of the switch and the nodes. The Management Information Bases implemented are for example : a MIB switch, as the known Bridge-MIB, providing in the agent module information about the switch such as the port numbers and the node identifiers and providing to the manager module information to request the agent module, a MIB interface as the known RFC1213-MIB or IF-MIB providing in the agent module the port status and providing in the manager module information to request the agent module. Other MIB may be implemented in the agent module and used by manager modules. In the switch, the agent module 130-A implements these MIB and stores the information of these implemented MIB in a memory 107.

In an embodiment of the invention, the manager module 130-M is thus arranged to retrieve node localization information from requests to the agent module 130-A, to store these information in a memory in a form of a table as described in figure 5 thus providing a node localization in the group of nodes and to update said table on change indication from the agent module.

The manager module 130-M sends a request for a connection with the agent module 130-A of a switch, this request identifying the switch, e.g. providing the IP address of the switch. This request is also a request for a session to be opened, e.g. an snmp-session identifying the switch. Different connections from a manager module 130-M may be requested in parallel. Once the connection is established between the manager module 130-M and the agent module 130-A, the manager module sends a *get-request()* function to the agent module of the switch in the cluster. In the node having the manager module, a user or a program (probe) defines the variable requested in this *get-request()* function, this variable may be the port number. An agent module retrieves the port number in its memory 107 of figure 4 and sends it to the manager module. A user or a program (probe) in the manager module may also request for the status of the port having this port number. The port is identified with its port number indicated in the *get-request()* function as an input variable. An agent module retrieves the status of a port in its memory 107 of figure 4, the port status of the switch being stored in the memory 107 when implementation of MIB interface. The variable *port status* is indicated in a return *get-response()* function and may have value *down* or *up*. The port

α

status *down* indicates that no node has sent a signal or message to this port, so there may be no node connected to the port or the connected node may not be alive (dead). The port status *up* indicates that a node has sent a signal or message to this port, so a node is connected to the port. The port status *up* in the *get-response()* function may be completed with a value *learned* indicating that the port is connected to a node whose identifier is in the memory 107 of the switch, accessible for the agent module.

In this last case, the manager module requests the agent module for the identifier of the node connected to the port having the *learned* value. The manager module requests for this identifier using another *get-request ()* function specifying the variable *node identifier* connected to the port having the given port number. If the agent module retrieves the node identifier of this node and sends it back to the manager module in the variable *node identifier* using *get-response ()* function, the manager module can retrieve the data couple "port number/node identifier". The manager module stores this data couple in a list which may be a table T in memory.

In another embodiment, a table of port numbers in the agent module may also have been requested by the manager module with the function of exhibit I-3-a identifying the snmp_session. A table of port status in the agent module may have been requested by the manager module with the function of exhibit I-3-b identifying the snmp_session and a table of node identifiers in the agent module may have been requested by the manager module with the function of exhibit I-3-b identifying the snmp_session. In this case, the manager module has retrieved all the information from the agent module, may then process each port number, port status and node identifier to establish the table T.

In an embodiment of the invention and by way of example of figure 5, the table comprises a first column C1 defining the port identifier (P-ID) being e.g. the port number and a second column C2 defining the identifier of the node (C-ID) connected to this port, the identifier of the node being e.g. the ethernet address for an ethernet switch and an IP address for an IP switch. In an embodiment, the table only indicates the ports being connected to an identified node.

The table is advantageously updated on agent module's message called *trap(code)* indicating an exceptional event such as :

- the port status has changed to *down*,
- the port status has changed to *up*.

The *trap(code)* provides the internet switch address and the port identifier (e.g. port number) for which the status has changed. The manager may request more information on the basis of the *trap()* agent module's messages.

In an embodiment of the invention, a method is also described hereinafter.

Thus, figures 6 illustrates a method to build the table T according to an embodiment of the invention based on manager module requests. Figure 7 illustrates a complementary method to update data couple of the table T of figure 5.

In figure 6, the process is aimed to build a table of at least data couples indicating port identifier/node identifier. In operation 601, the manager module requests an agent module of a switch designated with its identifier (e.g. IP address of the switch or the name of the switch) for the status of a given port designated with its port identifier (e.g. its port number or its MAC address). At operation 602, the agent module having retrieved this port status, e.g. in a database of the memory 107, sends the port status and other additional information to the manager module of the requesting node.

If the port status indicates that a node is connected to this port and that its node identifier is known at operation 604 ("learned"), the manager module may request the agent module to determine the identifier of the connected node at operation 608. The agent module may retrieve this information in a Management Information Base implemented in the agent module as hereinbefore described and sends it to the manager module. At operation 610, the manager module retrieves the node identifier corresponding to the port number and stores the data couple in a table, this data couple indicating at least the node identifier and the port identifier. At operation 610, a data couple corresponding to the same port identifier may be already stored in the table. In this case, the retrieved node identifier (new node identifier) and the node identifier already stored in the table (old node identifier) are compared and responsive to a difference between them, the old node identifier is replaced by the new node

α

identifier : the data couple in the table is thus updated. If other ports may be requested by the manager module at operation 612, the process returns to operation 601, else it ends.

If the port status indicates that the port is down, or if the port status indicates that a node is connected to this port and without indicating that the node identifier is known (or indicating that the node identifier is not known) at operation 604, the manager module may restart operations 601 to 604 for this port. The manager module restarts operations 601 to 604 for this port until the port status is *up* at operation 604 or until at operation 605 the manager module has restarted R consecutive times operations 601 to 604 for this port, R being an integer greater than 1. In this last case, the flow-chart continues at operation 612.

The flow chart of figure 6 may be repeated regularly to request for node identifier connected to a port identifier in order to update the table and to maintain a current table.

A manager module may execute the flow-chart of figure 6 in parallel for different ports in an agent module or in several agent modules.

In figure 7, a modification of the status of a port may appear in the switch, that is to say, a node having a down status may change to an up status and reciprocally. In this case, an agent module sends a trap() function, as described hereinbefore, in operation 702. The manager module receives then this trap at operation 704. If the port status indicates the value *up*, at operation 710 the flow-chart continues in figure 6 operation 601. For an already stored data couple in the manager module's memory, the manager module retrieves the node identifier for the port and updates the already stored data couple in operation 610 of figure 6. If the port status indicates the value *down* at operation 706, the data couple in the manager module's memory is invalidated at operation 708. After operations 708 or 710, the flow-chart ends.

The invention is not limited to the hereinabove embodiments. Thus, the table of the manager module's memory may comprise other columns or information concerning for example the time at which the information for the port and the connected node is retrieved. The manager module may regularly request for information such as the port status and the node identifier connected to this port. The manager module may define a period of time to retrieve said

information. In an embodiment, the table may also indicate all the ports and their status. If the node has a down status or if it is not identified, the column C2 is empty. This enables the manager module, the node having the manager module, or a user requesting this node, to have a sort of map for ports of a switch and to know to which port the node is connected.

If the port of a node is down, this port status is indicated in the table and the node connected to this port may be changed and may be connected to another port having an *up* status.

The invention covers a software product comprising the code used in the invention, specifically in the manager module.

The invention also covers a software product comprising the code for use in the method of managing a node localization.

α

Exhibit I

SNMP messages

I-1- Messages from the manager module

get-request(var, [,var,...])

get-next-request(var, [,var,...])

set-request(var, val,[,var, val...])

I-2-Messages from the agent

get-response(var, [,var,...])

trap(code)

I-3- Examples of Messages from the manager module

I-3-a snmp_get_port_table (struct snmp_session*ss)

I-3-b snmp_get_oper_status (struct snmp_session*ss)

I-3-c snmp_get_fdb_table (struct snmp_session*ss)

α

Claims

1. A distributed computer system, comprising :

- a plurality of nodes (Ni),
- a switch (SI) having ports (P),
- each port (P) having an identifier (P-ID),
- each node having an identifier (C-ID) and some nodes being connected to ports of the switch,
- the switch comprising an agent code (130-A) arranged for accessing to port status and identifiers of nodes (C-ID) connected to ports,
- a node (Ni) in said plurality comprising a manager code (130-M) arranged for
    - retrieving the status of a port from the agent code (130-A) and, responsive to said status meeting a given condition, requesting the agent code (130-A) for the identifier of the node (C-ID) connected to said port,
    - maintaining a table (T) of data groups comprising identifier of a node (C-ID) connected to a port and the identifier of said port (P-ID).

2. The distributed computer system of claim 1, wherein the manager code (130-M) is arranged to request the agent code (130-A) for status of ports and, responsive to said status meeting a given condition, requesting the agent code (130-A) for identifiers of the nodes connected to said ports.

3. The distributed computer system of any of the preceding claims, wherein the status of a port indicates that the port is up or down.

4. The distributed computer system of any of the preceding claims, wherein the status of a port indicates, when the port is up, that the node identifier is known or unknown.

5. The distributed computer system of any of the preceding claims, wherein the given condition comprises that the port status is up and indicates that the node identifier (C-ID) is known.

∝

6.The distributed computer system of any of the preceding claims, wherein the agent code (130-A) is arranged to send a message indicating a new port status and, the manager code (130-M) is arranged

- if the port status is down, for invalidating a data group in the table (T) having the same port identifier,
- else, responsive to said status meeting the given condition, for requesting the agent code for the identifier of the node connected to said port.

7. The distributed computer system of any of the preceding claims, wherein the manager code is arranged for comparing, for the same port identifier, the requested node identifier and the node identifier in the table (T) and responsive to a difference between the requested node identifier and the node identifier in the table, updating the node identifier for the port identifier in the table.

8. The distributed computer system of any of the preceding claims, wherein the port identifier (P-ID) is a port number.

9. The distributed computer system of any of the preceding claims, wherein the group of data in the table (T) comprises the time for the storage of the port and the node identifiers.

10. The distributed computer system of any of the preceding claims, wherein the manager code (130-M) is arranged for repeating regularly to request the node identifier of a port identifier.

11. A method of managing a localization of distributed computer system comprising a plurality of nodes and a switch having ports, each port having an identifier, each node having an identifier and being connected to a port of the switch, the switch having access to port status and identifiers of nodes connected to ports, comprising the following steps from said node:

a. retrieving the status of a port from the switch (601,602;704)

b. responsive to said status meeting a given condition (604;706), retrieving the identifier of the node connected to said port from the switch (608), and

α

c. maintaining a table of data groups comprising port identifier and the identifier of the node connected to said port (610;708).

12. The method of claim M1, wherein step a. comprises requesting the agent code for status of ports (601).

13. The method of any of the preceding claims, wherein step b. comprises, responsive to said status meeting a given condition (604), requesting the agent code for identifiers of the nodes connected to said ports (608).

14. The method of any of the preceding claims, wherein the status of a port of step a. indicates that the port is up or down.

15. The method of any of the preceding claims, wherein the status of a port of step a. indicates, when the port is up, that the node identifier is known or unknown.

16.The method of any of the preceding claims, wherein the given condition of step b. comprises that the port status is up and indicates that the node identifier is known (604).

17. The method of any of the preceding claims, wherein step a. comprises receiving a message from the switch indicating a new port status (704) and step b. comprises

- if the port status is down (706), invalidating a data group in the table having the same port identifier (708),
- else, responsive to said status meeting the given condition (706, 604), requesting the agent code for the identifier of the node connected to said port (608).

18. The method of any of the preceding claims, wherein step c. comprises comparing, for the same port identifier, the requested node identifier and the node identifier in the table and responsive to a difference between the requested node identifier and the node identifier in the table, updating the node identifier for the port identifier in the table.

19. The method of any of the preceding claims, wherein the port identifier is a port number.

α

20. The method of any of the preceding claims, wherein the group of data in the table comprises the time for the storage of the port and the node identifiers.

21. The method of any of the preceding claims, wherein step a. to step c. is repeated regularly to request for node identifier connected to a port identifier and to update the table.

22. A software product, comprising the code used in the distributed computer system as claimed in any of claims 1 through 10.

23. A software product, comprising the code for use in the method of managing a distributed computer system as claimed in any of claims 10 through 21.

x (17 pages) CABINET NETTER

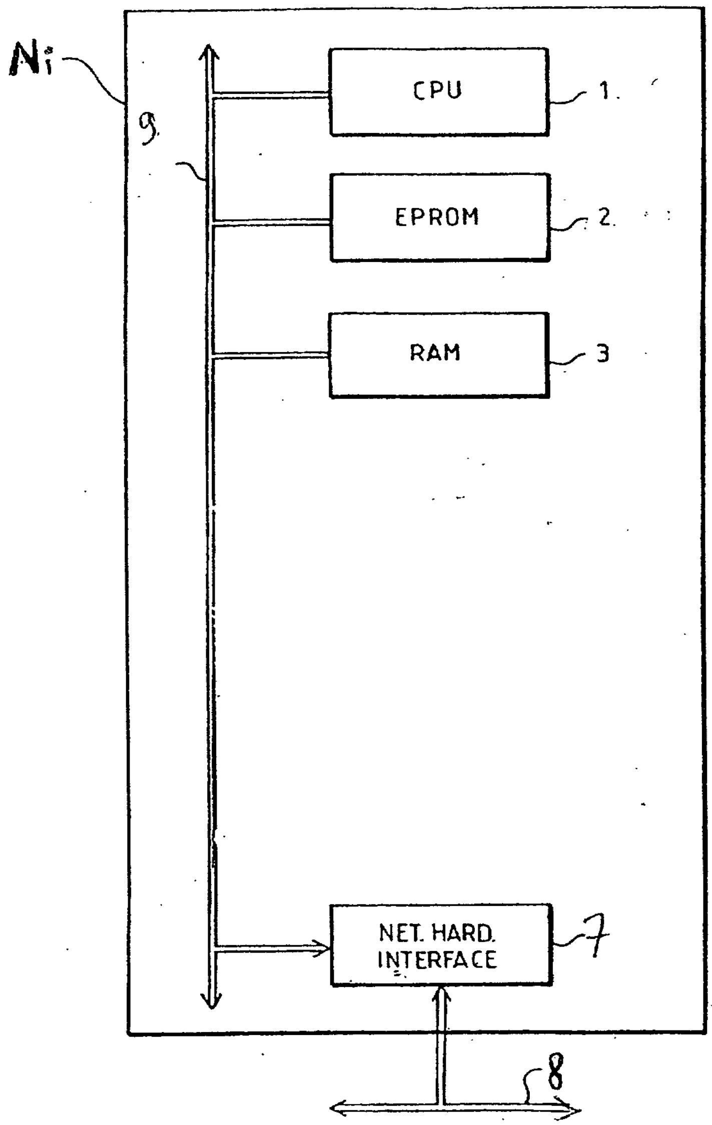


FIG 1

CABINET NETTER

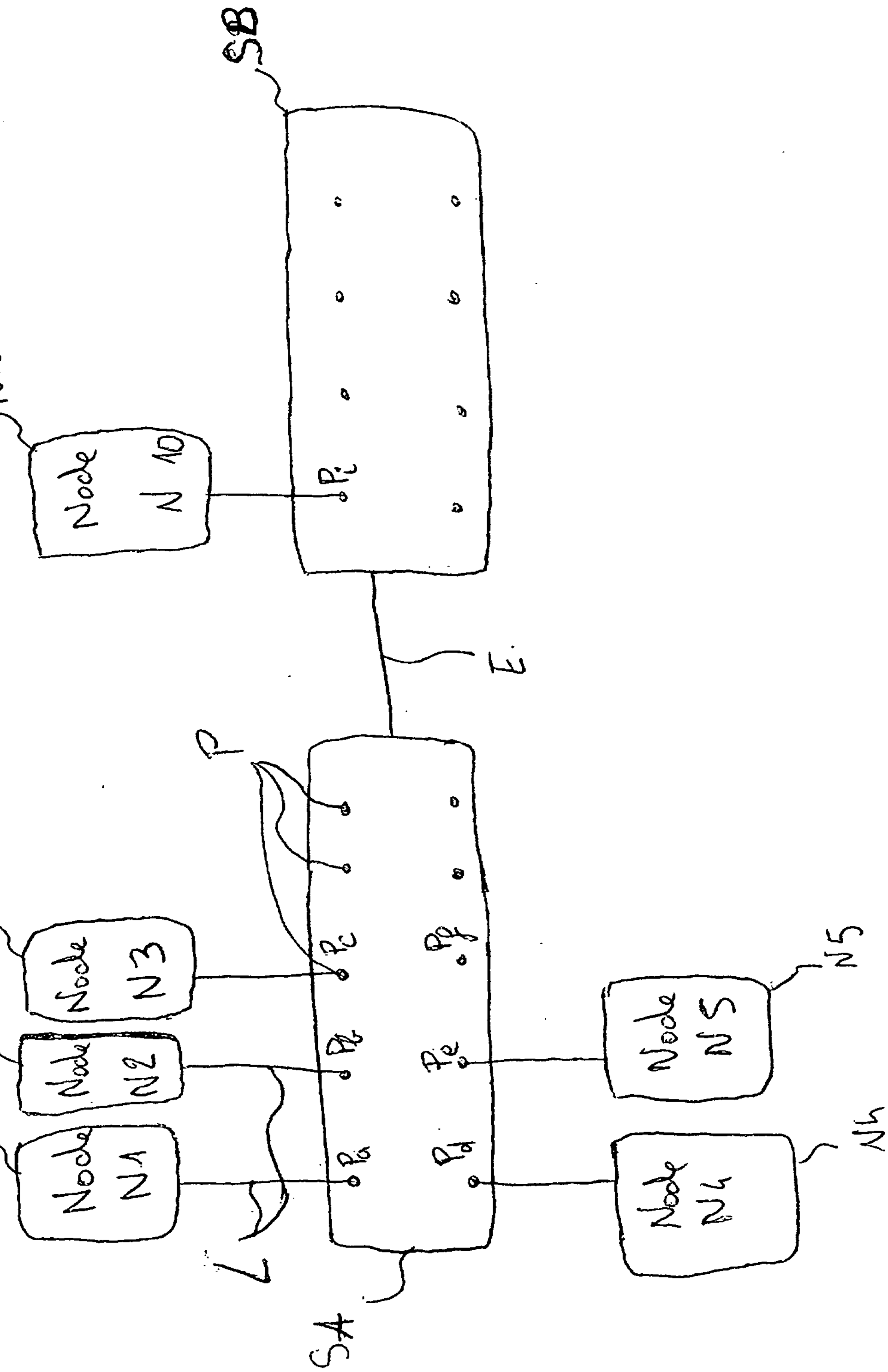


FIG 2

CABINET NETTER

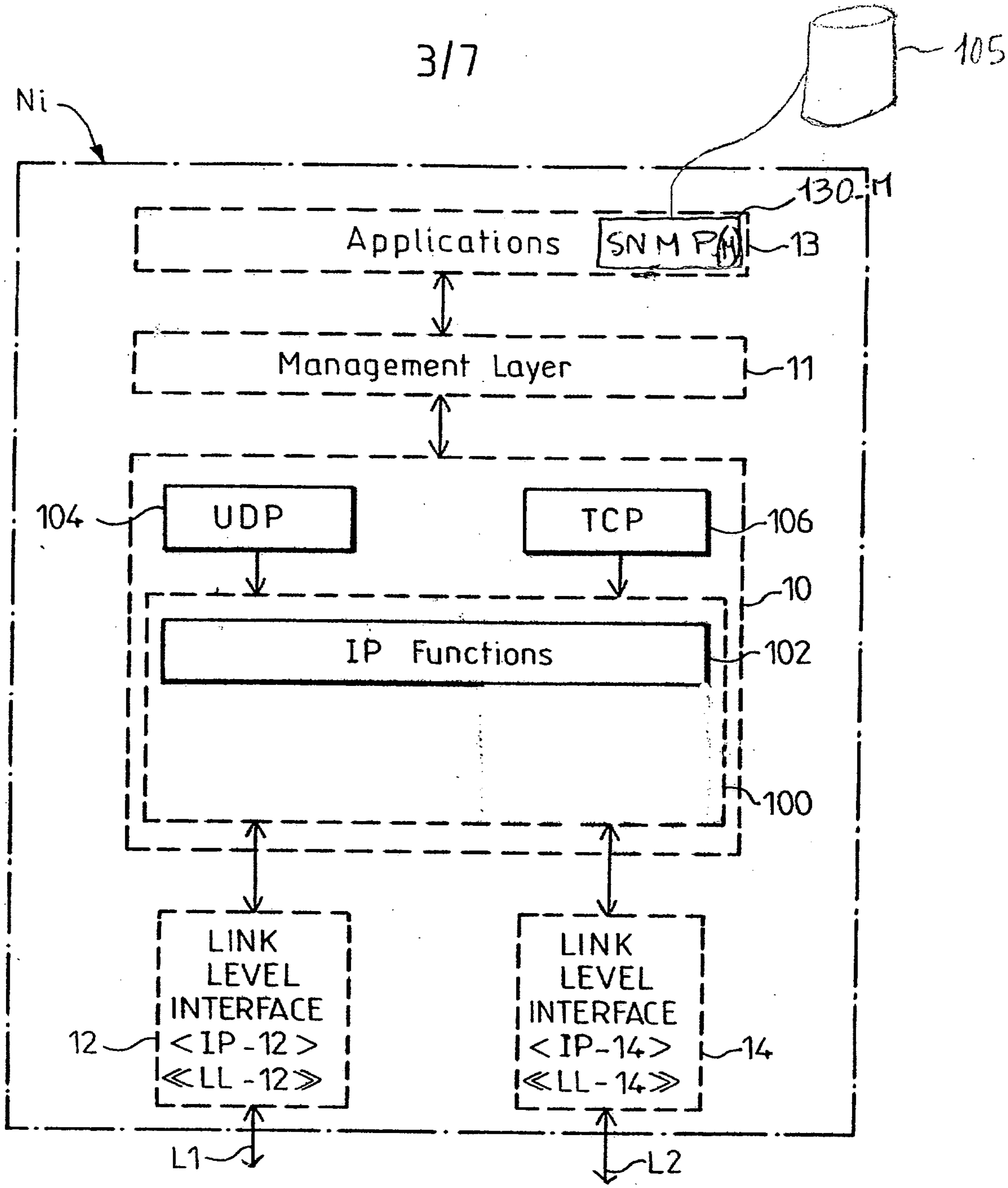


FIG. 3

CABINET NETTER

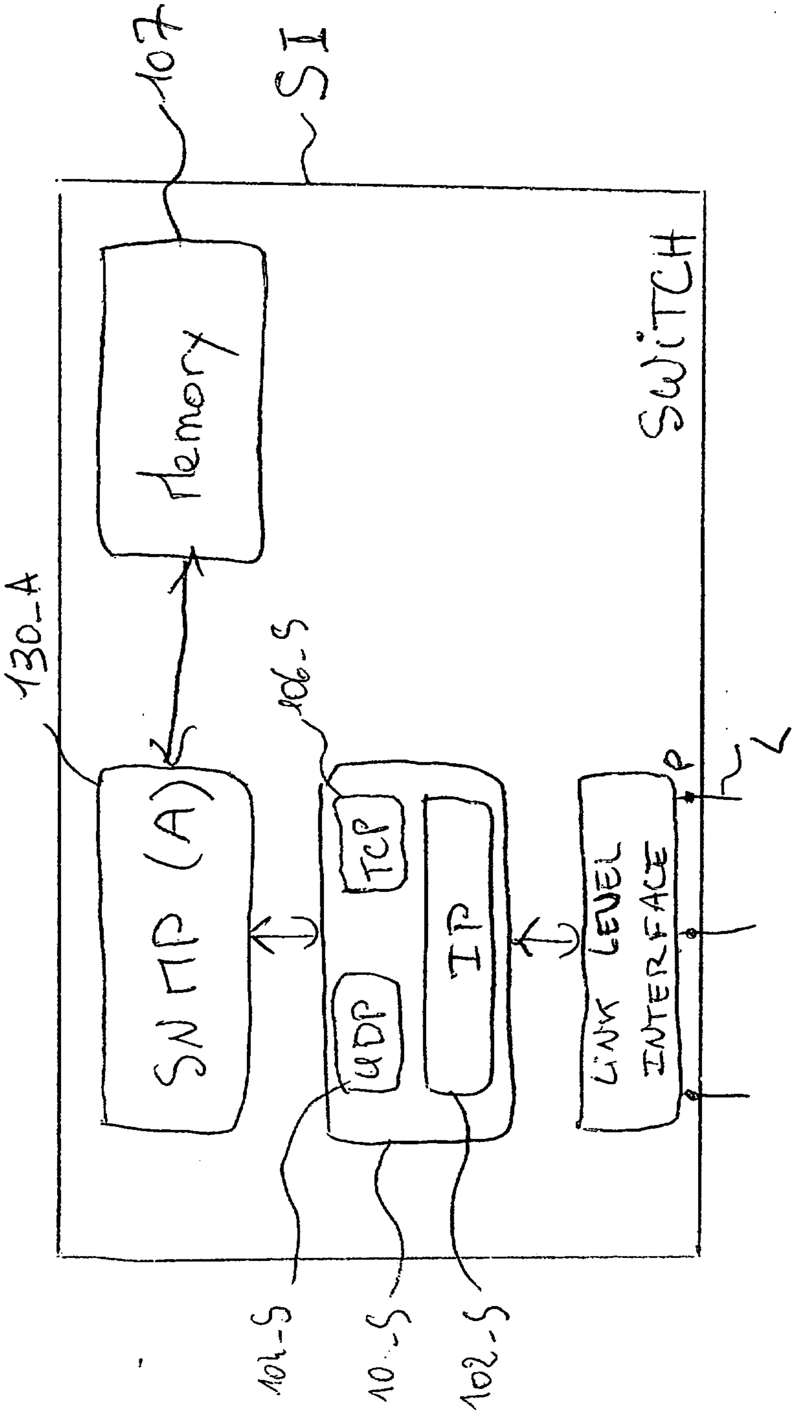


FIG 4

CABINET NETTER

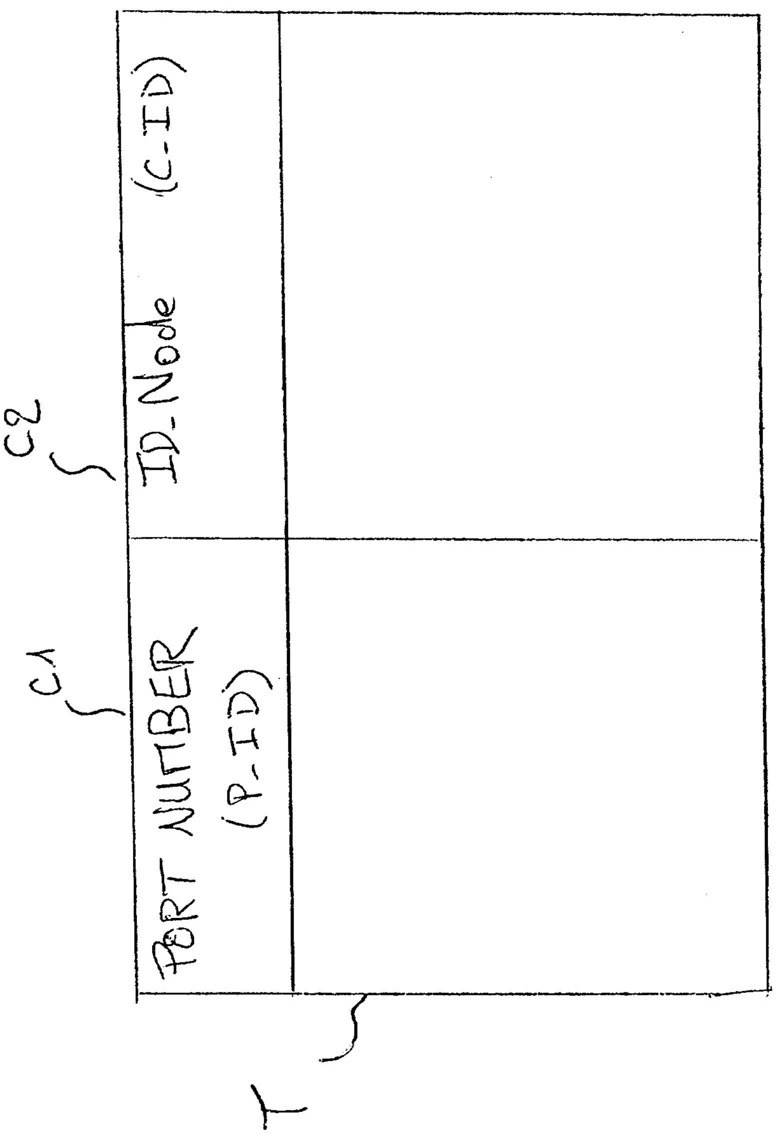
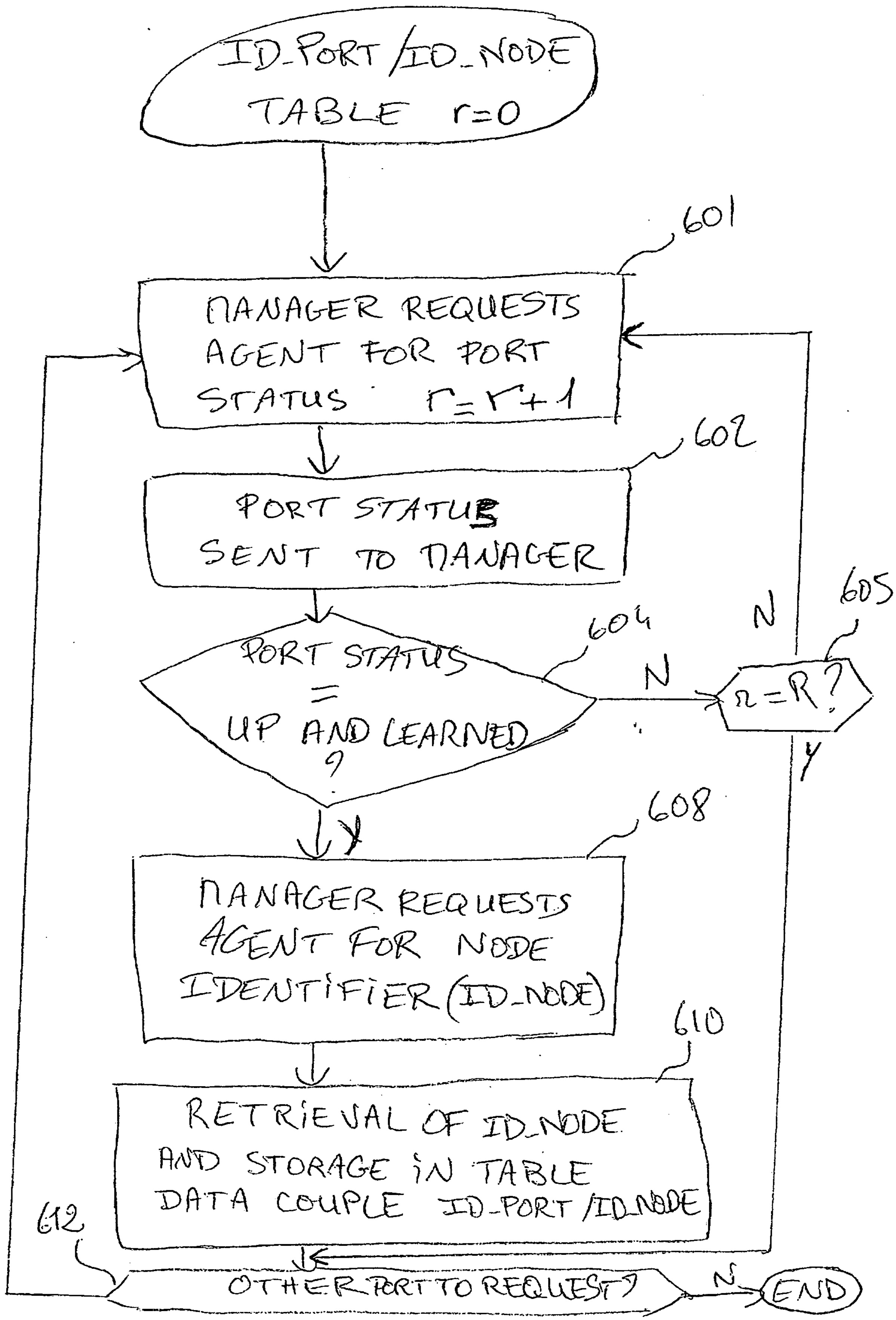


FIG 5

CABINET NETTER

FIG6



CABINET NETTER

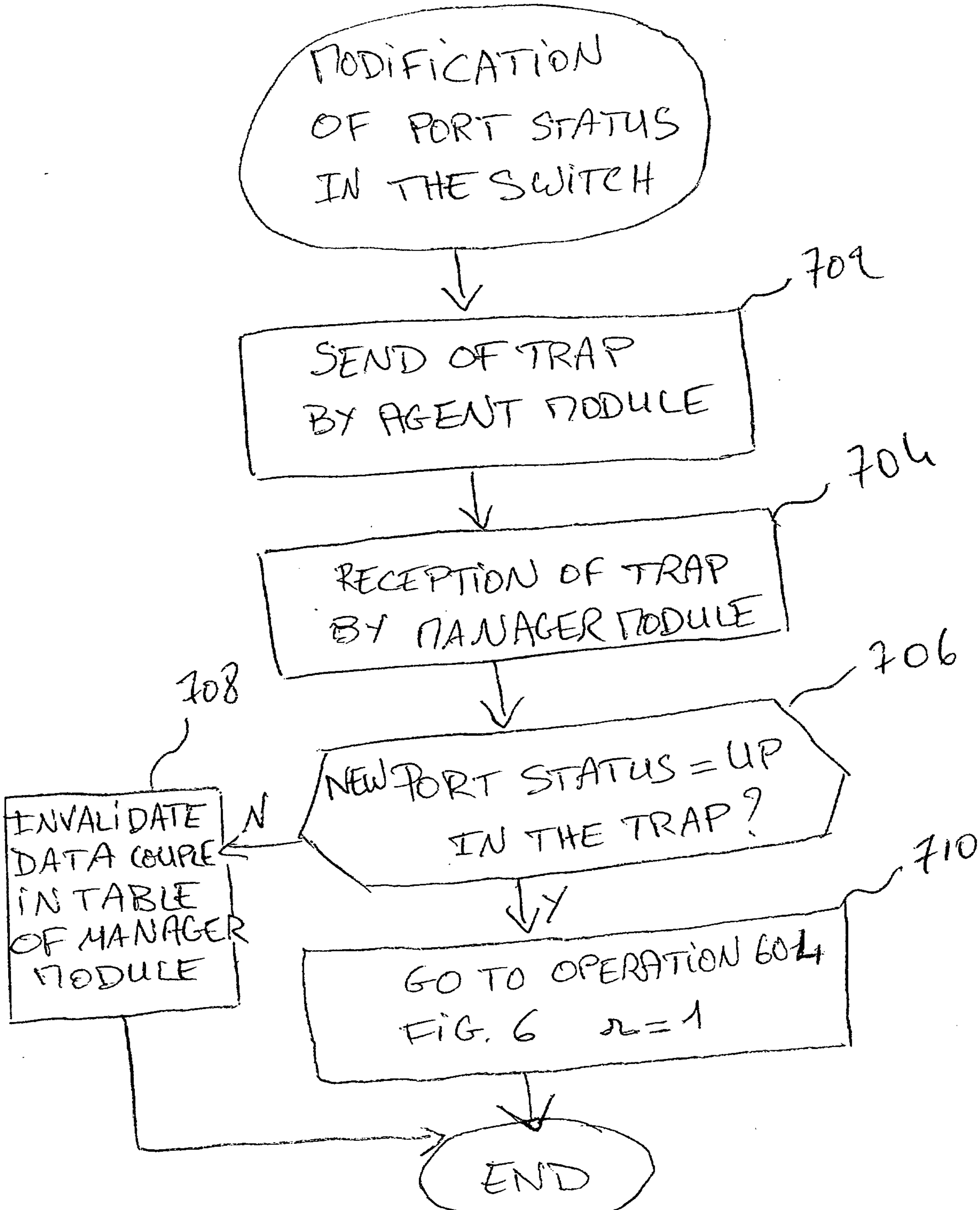

MODIFICATION OF PORT STATUS IN THE SWITCH
702
SEND OF TRAP BY AGENT MODULE
704
RECEPTION OF TRAP BY MANAGER MODULE
706
NEW PORT STATUS = UP IN THE TRAP ?
N
708
INVALIDATE DATA COUPLE IN TABLE OF MANAGER MODULE
Y
710
GO TO OPERATION 604 FIG. 6 n=1
END

FIG 7